НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»

Теплоенергетичний факультет

Кафедра автоматизації проектування енергетичних процесів і систем

До захисту допущено

Завідувач кафедри

_____ О.В. Коваль
(підпис) (ініціали, прізвище)

“___”_____2019р.

Дипломна робота

на здобуття ступеня бакалавра

з напряму підготовки 6.050103 “ Програмна інженерія “

на тему “Серверний застосунок стенду для моделювання віддаленого моніторингу та управління технологічними процесами“

Виконав (-ла): студент (-ка) ___4___курсу, групи_ТВз-51_

Гуцол Дмитро Ігорович _____
(прізвище, ім’я, по батькові) (підпис)

Керівник доцент Ковальчук А. М. _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Рецензент к.т.н., доцент Баранюк О. В. _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2019

Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет теплоенергетичний

Кафедра автоматизації проектування енергетичних процесів і систем

Рівень вищої освіти перший , бакалаврський

Напрям підготовки 6.050103 “ Програмна інженерія”

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ О.В. Коваль
(підпис)
”___”_____2019р.

ЗАВДАННЯ
на дипломну роботу студенту

_____
(прізвище, ім’я, по батькові)

1. Тема роботи “Серверний застосунок стенду для моделювання віддаленого моніторингу та управління технологічними процесами“
_____

керівник роботи Ковальчук Артем Михайлович, доцент
(прізвище, ім’я, по батькові науковий ступінь, вчене звання)

затверджена наказом вищого навчального закладу від ”__”____ 201__р. № __

2. Строк подання студентом роботи _____

3. Вихідні дані до роботи Мікрокомп’ютер, мікроконтролер, кінцеві модулі, екран, керуючий модуль, середовище розробки Visual Studio 2017, мова програмування C++.

4.Зміст розрахунково-пояснювальної записки (перелік завдань, які потрібно розробити) Проаналізувати існуючі рішення, спроектувати за зібрати стенд, налаштувати усі складові стенду, розробити серверний застосунок для нього

5. Перелік ілюстративного матеріалу
1. Актуальність теми. 2. Задача реалізації системи. 3.Аналіз існуючих систем. 4. Реалізація апаратної частини стенду. 5. Програмні засоби. 6. Реалізація серверного за стосунку. 7. Висновки.
_____
_____
_____
_____

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання ”_10_”___жовтня_____2018р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітки
1.	Вивчення та аналіз задачі	01.12.2018-31.03.2019	
2	Розробка архітектури та загальної структури системи	01-21.04.2019	
3.	Розробка структур окремих підсистем	22-28.04.2019	
4.	Програмна реалізація системи	29.04-13.05.2019	
5.	Оформлення пояснювальної записки	06.05-01.06.2019	
6.	Захист програмного продукту	25.05.2019	
7.	Передзахист	01.06.2019	
8.	Захист	17.06.2019	

Студент _____ Гуцол Д. І.
(підпис) (прізвище та ініціали,)

Керівник роботи _____ Ковальчук А. М.
(підпис) (прізвище та ініціали,)

АНОТАЦІЯ

Метою роботи була реалізація серверного застосунку стенду для моделювання віддаленого моніторингу та управління технологічними процесами. Було зпроектовано, запрограмовано та налаштовано програмно-апаратну складову стенду. Після цього була реалізована серверна частина. Однією з переваг проекту є можливість додавати або забирати окремі модулі стенду, що робить його функціонал гнучким та надає можливість підлаштуватись до вимог користувача. Система збирає і обробляє технологічні дані та працює в реальному часі. Необхідну користувачу інформацію можна подивитись на екрані самого стенду та за допомогою спеціальної web-сторінки зі зручним та зрозумілим інтерфейсом або мобільного додатку.

Записка містить 43 сторінки, 13 рисунків, 16 посилань та 3 додатки

ABSTRACT

The purpose of the work was to implement a server application for the stand for simulation of remote monitoring and control of technological processes. The software and hardware components of the stand were programmed, programmed and configured. After that, the server part was implemented. One of the advantages of the project is the ability to add or remove individual booth modules, which makes its function flexible and provides the ability to adapt to user requirements. The system collects and processes technological data and works in real time. The information you need for the user can be seen on the screen of the stand and using a special web-page with a convenient and understandable interface or mobile application.

The note contains 43 pages, 13 drawings, 16 links and 3 attachments

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕМ, СКОРОЧЕНЬ І ТЕРМІНІВ

ООП — об’єктно-орієнтоване програмування

STL — стандартна бібліотека шаблонів

СРБ — система розумного будинку

VS — Visual Studio

UART — Universal asynchronous receiver/transmitter

ЗМІСТ

ВСТУП

Швидкий розвиток сучасного світу та суспільства в цілому потребують інноваційних технологій, нових систем та розробок, що будуть впливати на всі сфери людського життя та діяльності людини. Спрощення технологічного процесу, покращення умов праці, проживання, комфорту в різних галузях життєдіяльності також є метою створення стенду моделювання віддаленого моніторингу та управління технологічним процесом. Це стає просто необхідністю.

Яскравим прикладом використання даної технології є розумний будинок. Зараз це явище набуває великої популярності. Розумний будинок – це будинок або будь-яке приміщення, яке використовується людьми. У них наявні системи забезпечення, а також усі електроприлади, якими можливо керувати за допомогою пульта-дисплею, пов’язані між собою.

Невід’ємною частиною розумного будинку є можливість управляти різними приладами одночасно за допомогою спеціального додатку на власному мобільному пристрої або web-сайту. Використання цієї функції повинне бути абсолютно зручним та зрозумілим.

Другою невід’ємною частиною розумного будинку є можливість віддаленого моніторингу. За допомогою цієї технології можна в реальному часі у будь-якому куточку світу, де є доступ до інтернету, переглянути необхідну інформацію з різноманітних показників та датчиків, вмонтованих у будинку чи приміщенні. Наприклад, моніторинг та управління температури повітря, рівню вологості, освітленості, затоплення, датчиків руху, сигналізації, гучності звукових пристроїв та іншого.

У цій роботі було зпроектовано, зібрано та налаштовано стенд моделювання віддаленого моніторингу та управління технологічним процесом. Було обрано, куплено та запрограмовано мікрокомп’ютер та мікроконтроллер. За допомогою цього стенду можна зрозуміти основні принципи роботи системи розумного будинку та виконувати функції управління технологічним процесом.

Також, відповідно, у роботі була реалізована програмно-апаратна частина. Основними задачами були швидкість взаємодії між собою усіх компонентів стенду, їх справна робота та, відповідно, написання серверної частини.

У записці містяться .. розділів.

Перший розділ включає..

Другий розділ включає..

…

1. ПОСТАНОВКА ЗАДАЧІ СЕРВЕРНОГО ЗАСТОСУНКУ СТЕНДУ ДЛЯ МОДЕЛЮВАННЯ ВІДДАЛЕНОГО МОНІТОРИНГУ ТА УПРАВЛІННЯ ТЕХНОЛОГІЧНИМИ ПРОЦЕСАМИ (РОЗУМНИМ БУДИНКОМ)

1.1 Актуальність теми

У наш час людство дуже швидко розвивається. З ним розвиваються і технології. Віддалений моніторинг та управління технологічними процесами стає просто необхідністю. Вони дають можливість керувати процесом та виконувати робочі обов’язки на відстані. Така функція може значно підвищити ефективність роботи та полегшити її виконання. Ці системи можна використовувати у багатьох сферах життєдіяльності людини: будівництво, управління підйомними кранами, гаджетами, відеокамерами, виробництво, харчова промисловість, машинобудування, та багато іншого, зокрема розумний будинок.

Розумні будинки зараз стають все актуальніші, розвиваються, змінюються та вдосконалюються. Адже ця система має багато переваг: зручність та простота у використанні, економія електроенергії, опалення та просто часу користувача, контроль мікроклімату, музики та усіх електроприборів, якими можна управляти за допомогою пульту. Також вона забезпечує охорону та безпеку будинку. Використовувати всі ці функції можна голосовим управлінням.

Застосування технологій віддаленого моніторингу та управління технологічними процесами є просто необхідністю у розумних будинках.

1.2 Мета, завдання та вимоги роботи

Метою дипломної роботи є розробка апаратно-програмної частини стенду для моделювання віддаленого моніторингу та управління технологічними процесами. Необхідно наглядно та зрозуміло продемонструвати схему роботи такої системи.

Основними завданнями роботи є:

1. Реалізація серверного застосунку стенду

2. Проектування, виготовлення та програмування апаратної частини стенду.

3. Налагодження справної роботи стенду та взаємодії між собою його апаратних складових

Вимоги до роботи:

1. Робота в реальному часі

2. Швидка взаємодія між собою апаратних елементів стенду

3. Можливість додавати або забирати окремі модулі до стенду

4. Надійність конструкції

У системі розумного будинку(рисунок 1.1) зазвичай у використовуються хоча б декілька різних функцій. Наприклад автоматичне відкриття воріт, система

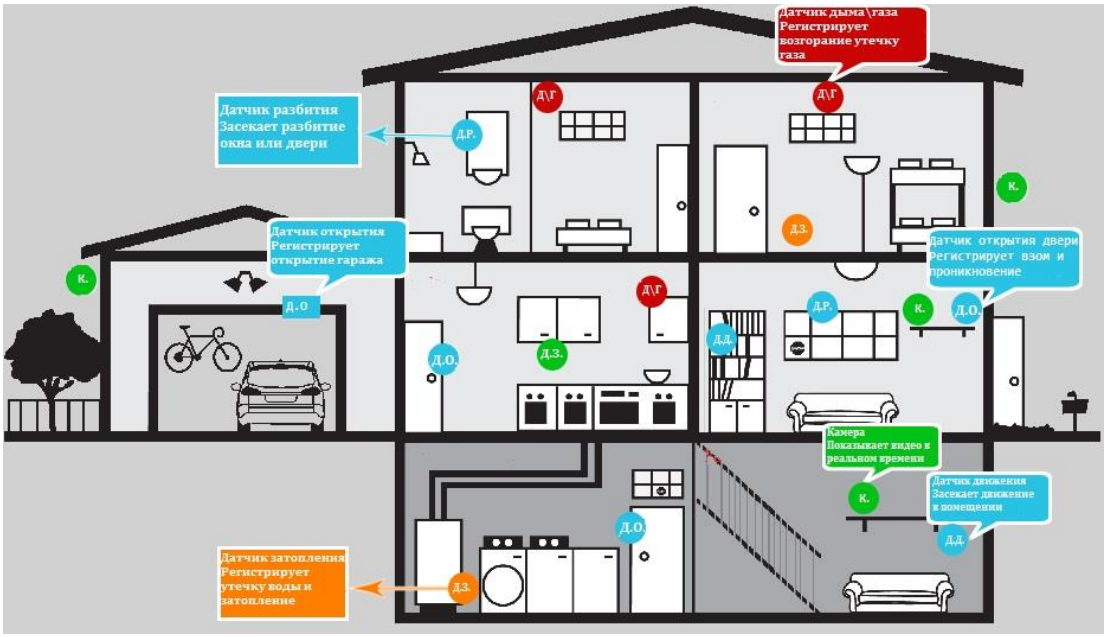

Рисунок 1. 1

охорони, відео нагляду та інші. Тому було вирішено зробити стенд з модульною

конструкцією. Тоді до нього можна буде додавати будь-які модулі з різними функціями та навпаки, забирати непотрібні. Таким чином система буде універсальнішою та гнучкішою.

Необхідно також визначитись з методом роботи апаратної частини. Після цього правильно підібрати усі його елементи та налагодити зв'язок між ними.

Програма повинна приймати інформацію з модулів для віддаленого моніторингу через гаджет та обробляти команди, надіслані з мобільного додатку або web-сайту для управління технологічними процесами та їх регулювання.

2. ОПИС ПРЕДМЕТНОЇ ОБЛАСТІ

Під час виконання цієї роботи було розглянуто системи розумних будинків та проаналізовано способи та можливості застосування стенду для моделювання таких задач. Системи віддаленого моніторингу та управління існують та широко використовуються не так давно. Проте вони дуже швидко розвиваються і вже зараз існує безліч різних технологій та компаній, які займаються розробкою, виготовленням та встановленням необхідного обладнання. Незважаючи на це, існуючих рішень схожого саме стенду для моделювання моніторингу та управління технологічними процесами знайдено не було. Тому було прийнято рішення обирати необхідні апаратні частини стенду опираючись на їх характеристики та функції. Для написання серверної частини було обрано мову програмування C++.

2.1 Опис існуючої системи моніторингу та керування віддаленими об’єктами регулювання

У цьому пункті розглянемо повноцінну систему віддаленого контролю. Її завданням є збір інформації, передачі даних, які будуть зберігатись та оброблятись пізніше. Її особливістю є модульна конструкція, яка може змінювати(додавати або забирати) певні функції та можливості.

Зазвичай, обробка вхідної інформації є основою таких систем віддаленого керування. Адже використовуючи ці дані, буде активуватись послідовність управління. Збір показників, їх аналіз та передача в систему управління. Ось що входить в поняття обробки даних.

Функціонування системи[2] забезпечують програмований мікропроцесор або відразу декілька пристроїв у зв’язку з певними засобами телемеханіки. Існує 2 типи передавання даних в залежності від потреб користувача. Місцеве, реалізоване

провідними типами зв’язку та віддалене, що використовують безпровідні рішення. У цій конкретній системі вирішили застосовувати протокол RS-232. Це дає змогу розширяти протокол. Зробити це можна використовуючи малодіючі модулі BlueTooth.

Дуже важливу роль у цій розробці відіграє універсальність її застосування. А саме повинна бути можливість її застосування у яких завгодно об’єкти. При цьому внесення змін у готову конструкцію повинні бути якомога менші. Конкретну систему можна поділити на 3 рівні:

1. Базовий. Він включає в себе мікропроцесор разом з індикаторами стану та має функцію вводу налаштувань вручну. Також доступна самодіагностика.

2. Безпровідна передача інформації. Тут повинні бути рідаомодеми, діапазон частот яких має значення 433 Мгц. Вони застосовуються для передачі необхідних параметрів.

3. Програмна обробка. На цьому рівні відбувається накопичування даних, їх безпосередня обробка за допомогою спеціальних інтелектуальних алгоритмів.

Зчитування даних та їх подальша обробка у даній системі відбувається за допомогою мікроконтролера Arduino mega 2560.

Даний мікроконтролер виконує такі важливі завдання як аналіз параметрів системи: процес передачі та прийому інформації між всіма складовими системи; стан ліній управління; напругу живлення. У випадку, коли мікроконтролер визначить якусь проблему, повідомлення про це та інформації про помилку передається на певний пункт обробки даних. До того ж, додатково активується сигналізація.

В алгоритмі роботи регулятора є функція періодичного активування головного контролера. Через це енергоспоживання не більше навіть під час роботи без застосування допоміжного контролера. У цій системі існують інтерфейси комунікації RS-232, які необхідні для налаштування зв’язку GSM модема або ж радіо модуля з регулятором. Він виконується на мікросхемі MAX-232. Вона ж, в свою чергу, змінює логічні стани +5 В та 0 у стани послідовного порта +-12 В мікроконтролера.

У системі присутній також роз’їм для пульта управління, на якому виведені всі параметри. У початковому режимі параметри можна лише подивитись. Проте існує функція ручного управління у другому режимі. Розробка друкованої плати відбувалась за допомогою середовища Sprint Layout. Це дає можливість змінювати конструкцію.

Контролер[15] у даній системі призначений для напруги від 0 до 5 В і роздільною здатністю 210 біт. Проте є можливість застосування передавачів інших типів за допомогою блоку перетворювачів. Так як систему потрібно час від часу перевіряти на справність, є переносний пульт управління. На ньому вказані точні показники стану. Оператор має можливість змінити ці показники на свої та провести діагностику.

Пульт підє’днується до системи через роз’їм. Коли він підключений, пульт використовує електроенергію. Тому, коли він відключений, блокується подача напруги на нього заради економії ресурсів.

2.2 Розумний будинок

Розумний будинок – це може бути будь-яке приміщення(магазин, будинок, офіс, тощо), облаштоване відповідною системою моніторингу та управління. Ця технологія набуває все більшої популярності через свої функції та переваги.

Система розумного будинку[1] забезпечує ряд можливостей для власника, що забезпечує комфортніше та економніше проживання. Так, у таких будівлях всі електроприлади, якими можна керувати за допомогою пульта-дисплею, пов’язуються між собою та можуть бути підключені до мережі. Це надає змогу керувати ними віддалено, за допомогою смартфона або іншого гаджету, що має доступ до інтернету. Всі прилади, що підключені до системи, також узгоджують всі свої дії між собою та оцінюють можливість виконання задачі, зважаючи на показники датчиків та обстановку.

Робота розумного будинку також економить ваші кошти. Оскільки будинок

оснащений багатьма датчиками, він може оцінювати отримані від них показники та, опираючись на задану програму, здійснювати певні дії, щоб зекономити електроенергію, воду, тощо.

2.3 Переваги використання розумного будинку

Основні переваги встановлення та використання розумного будинку:

1. Комфорт. Використання системи розумного будинку значно покращує та забезпечує комфортне проживання вдома. Адже щоб увімкнути світло, закрити штори, включити музику, телевізор або навіть все разом, вам потрібно просто натиснути кнопку на екрані смартфона.

2. Встановлення та економія. Замовити та встановити систему розумного будинку сьогодні дуже легко. Адже Компаній, які спеціалізуються на цьому на ринку вистачає. Це не займає багато часу та рахується неважкою процедурою. Функції, які дає ця система, допомагають економити на комунальних послугах в майбутньому до 30%. Здійснюється це, наприклад, таким чином: система автоматично вимикає світло у кімнаті, якщо вас там немає, вмикає режим «мінімум» на опалення до тих пір, поки ви не повернетесь

3. Безпека. Система розумного будинку забезпечує безпеку у домі. Оскільки там встановлені численні датчики, будинок «розуміє» обстановку всередині та зовні і може, при необхідності, виконати певні дії, задані програмою.

2.4 Основні функції розумного будинку

Система розумного будинку включає в себе багато різних функцій та можливостей. Їх можна використовувати у будь-якій кількості за бажанням та потребами користувача.

2.4.1 Пожежна безпека

Пожежна безпека виконує свої функції використовуючи датчики, які реагують на дим. Вони розташовані на всій території будинку таким чином, щоб у любому куточку вони могли зреагувати та передати сигнал. Також повинна бути тривожна кнопка.

Ще одна функція пожежної безпеки це система пожежогасіння. Так, у разі виникнення пожежі, вентиляція у місцях горіння автоматично відмикається, а система димовідводу, навпаки, повинна увімкнутись. Разом з нею спрацьовує і система пожежогасіння та розблоковуються усі двері в домі.

Пожежна сигналізація являє собою сукупність датчиків виявлення диму, загоряння, які встановлені спеціальним чином по всій території приміщення, ПЛК, який сприймає сигнал і передає його далі, а також сюди входить тривожна кнопка.

Пожежна сигналізація також включає систему пожежогасіння. Система вентиляції у місцях виникнення пожежі повинна відмикатись, повинна вмикатися система димовідводу, вмикається система пожежогасіння, відбувається автоматичне розблокування дверей . Все це складові системи пожежної сигналізації.

2.4.2 Система охорони

Охорона здійснюється за допомогою датчиків, які встановлені в таких місцях, у які можливе проникнення посторонніх, наприклад вікна або двері. Коли вони спрацьовують, вмикається світо-звукова сирена(сигналізація) і потім активується система відео-спостереження.

Ці системи потрібні для візуального нагляду за певним об’єктом. Відео-спостереження дає можливість одночасно дивитись за декількома об’єктами або за одним і ззовні, і всередині будинку. Система працює для того, щоб була можливість подивитись та перевірити обстановку у будинку, приміщенні тощо.

Найпростішою системою охорони є одна або більше камер разом з екраном, на якому можна буде дивитися відео з них. Відеокамери можуть кріпитися на рухомих пристроях на вулиці, так і у будинку. Так вони забезпечують цілодобовий нагляд за потрібним користувачу місцем.

Зазвичай разом із системою спостереження використовують й інші детектори, наприклад освітлення. На екрані монітору можуть відображатися зображення або зі всіх камер разом, або по черзі кожна.

2.4.3 Контроль доступу

Ще однією системою розумного будинку є контроль доступу. Її можна поділити на автономні і мережеві. Перші є найпопулярнішими у зв'язку зі своєю простотою і невеликій ціні. Зазвичай це набір з обладнанням, призначений для управління єдиними дверима. Він складається з:

1. Контролером зі збереженим на ньому кодами бейджів чи ключів.
2. Однієї клавіатури, що призначена для написання коду або зчитування ключів.
3. Доводчик, блок живлення та електричний замок.

Що стосується мережевих систем, то вони зазвичай створюються за принципом, суть якого полягає в управлінні доступом, використовуючи персональний комп'ютер. Так можна управляти системою допуску та застосовувати не однакові рівні допуску для тих самих користувачів, а також якісно контролювати усі приміщення та слідкувати за робочими годинами усіх співробітників.

Частіше всього, вони використовуються в комплексі системи безпеки якоїсь фірми. Там також застосовуються системи охорони та відео-спостереження за необхідним будинком тощо. Обладнання, яке використовується системах такого типу, частіше всього дуже різне та виконує різні функції.

2.4.4 Керування освітленням

Використання систем розумного світла найчастіше здійснюється за програмою. Вони дають можливість вигідно використовувати освітлювальні пристрої. Ці програми можуть бути активовані як в одній кімнаті, так і у всьому приміщенні. Це робиться дуже легко за допомогою панелі керування.

Розумне світло може використовувати задані програми та виконувати їх автоматично, зважаючи на обстановку. Наприклад, виходите з кімнати, а світло

вимикається там автоматично, і навпаки, вмикається, коли ви заходите. Ще однією можливістю розумного освітлення є здатність імітувати присутність людей вдома. Це може бути корисно, коли вас довго немає вдома, наприклад. Управління також може здійснюватись звичайними вимикачами.

2.4.5 Система оповіщення

Дана система самотужки виявляє деякі фактори, визначення яких запускає виконання певної послідовності дій, яка часто задається наперед. Це можуть бути системи голосового оповіщення, електронного, світлового тощо.

2.4.6 Вирішення побутових завдань

Ще одною перевагою розумних будинків є використання сучасного обладнання у домі з нею. Щоденні побутові завдання можуть виконуватися значно легше та швидше, якщо грамотно розприділити їх та користуватися розумним будинком. Наприклад, запустити пральну машину дистанційно перед тим, як йти додому з роботи. Це зекономить багато часу. Ще одним прикладом є автоматичний полив газону або закриття усіх вікон у домі під час дощу. Автоматичне вмикання кавової машини тоді, як ви увімкнете світло у спальні зранку тощо.

2.4.7 Контроль клімату

Ця система використовує датчики температури та вологості. Розташовані у кожній кімнаті, вони збирають дані та виконує необхідні дії, щоб виконати задану вами програму. Наприклад, підтримання комфортної температури у спальні, автоматичне зволоження повітря при певному рівні вологості, вимкнення опалення в будинку під час відсутності там людей та багато інших ситуацій, в яких допомогла б ця система.

2.4.8 Контроль води та газу

Контроль за протіканням води або газу дуже корисна система. Завдяки їй, завжди можна бути спокійним і не гадати, чи закрили ви воду перед виходом і не

боятися витоку газу. Адже ця система перекриє воду і газ автоматично при будь-якому прояві небезпеки. Також ви можете на відстані набирати або спускати воду з басейну або ванни, наприклад.

2.4.9 Контроль електроенергії

Система контролю за електроенергією може автоматично зупиняти подачу електроенергії в певну частину будинку або навіть розетку. Дуже зручна система, особливо для людей, які постійно забувають вимкнути, наприклад праску або кондиціонер. Задавши відповідну програму, система може вимикати електропостачання у всіх кімнатах або вибраних місцях крім того, де людина знаходиться в даний момент.

Висновки до розділу

У цьому розділі була розглянута система “розумного будинку”, а також її основні функції та переваги. Проаналізувавши отримані дані, можна зробити висновок, що створений стенд для моделювання віддаленого моніторингу та управління технологічними процесами чудово демонструє схему роботи такої система як “розумний будинок”.

3. АПАРАТНА ЧАСТИНА СТЕНДУ

При розробці апаратної частини стенду, дуже важливо правильно підібрати його складові, зробити схему і загальний проект.

3.1 Загальний опис стенду

При створенні апаратної частини (Рисунок 3.1) стенду для моделювання віддаленого моніторингу та управління технологічними процесами було проаналізовано принцип роботи системи розумного будинку, розглянуто актуальні варіанти всіх елементів, з якого він складається.

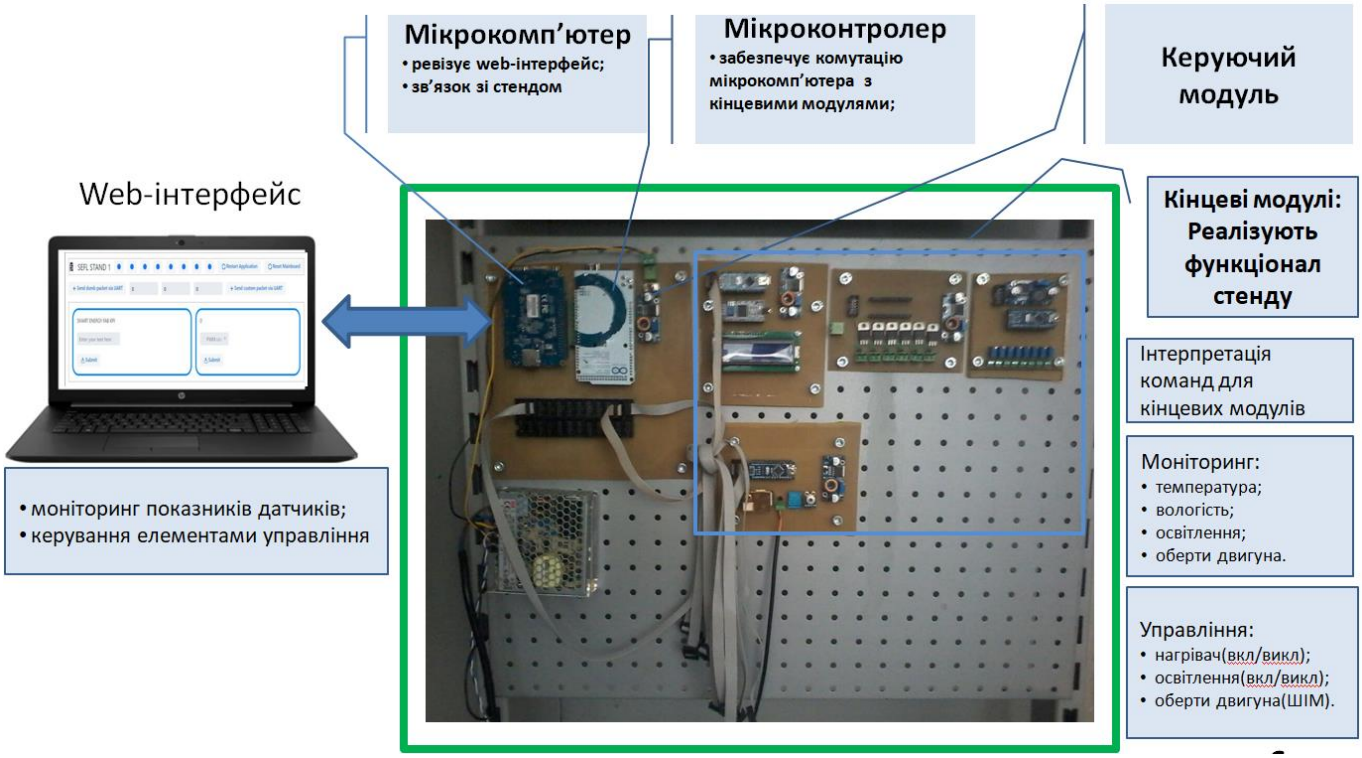

Рисунок 3.1 — складові стенду та їх функції

Після аналізу були обрані усі складові стенду та спосіб підключення їх між собою.

Коли всі елементи стенду було придбано, вирішено, який принцип роботи

буде реалізовано, потрібен був план подальших дій. Розрахувавши час та ресурси, він був складений і ми приступили до його виконання.

План був успішно виконаний і ми отримали закінчену конструкцію стенду (Рисунок 3.2) з усіма необхідними елементами.

Рисунок 3.2 — Готовий стенд

Стенд складається з мікрокомп’ютера, мікроконтролера, маленького екрану для виводу інформації, керуючого модуля, гнізда для підключення інших елементів та кінцевого модулю, який і реалізує весь функціонал стенду. Хотілося зробити його якомога гнучкішим в функціональному плані. Тому сама конструкція та принцип

роботи стенду були організовані таким чином, що до нього можна підключити до 20 окремих модулів, які можуть працювати одночасно.

Була створена схема (Рисунок 3.3) роботи стенду. В ній показані принцип роботи стенду, типи з'єднань та способи взаємодії між собою усіх його апаратних елементів.

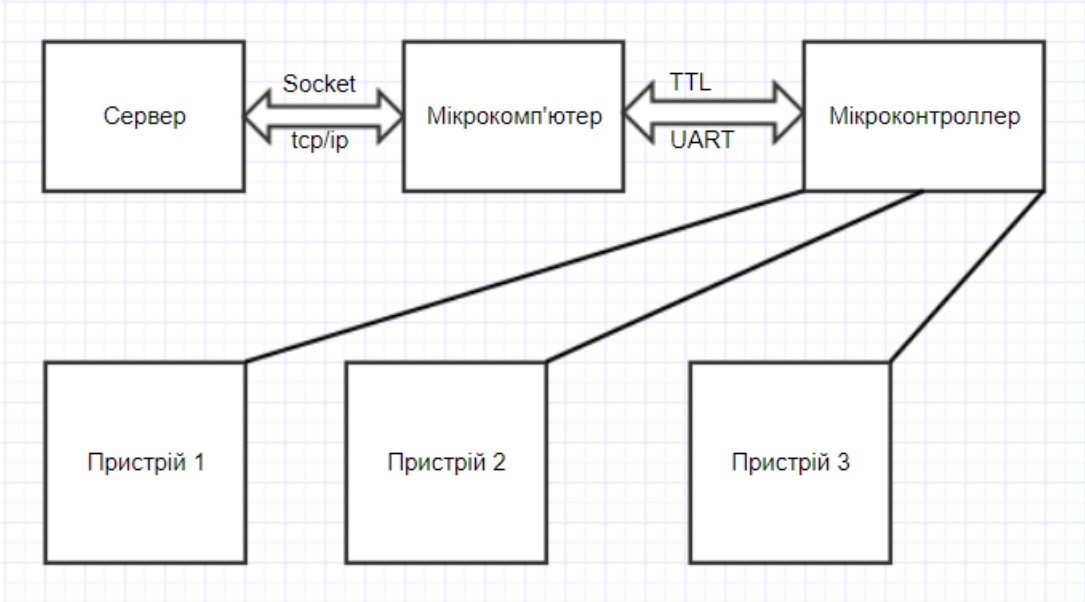

Рисунок 3.3 — схема роботи стенду

Так, мікрокомп'ютер забезпечує зв'язок будь-якого смартфону, ПК чи іншого пристрою з доступом в інтернет, зі стендом. Ще однією його функцією є ревізія web-інтерфейсу.

Мікроконтролер, в свою чергу забезпечує комутацію мікрокомп'ютера з кінцевими модулями(пристроями).

Усі дані можна вивести на екран стенду. Це можливо зробити зі спеціального додатку на смартфоні або використовуючи web-сайт. Там саме користувач може і переглянути всю інформацію, яку надає стенд.

До стенду можна підключити до двадцяти окремих пристроїв одночасно. Тому функціонал стенду може змінюватись при необхідності або бажанні користувача.Кінцеві модулі підключаються до гнізда, яке зв'язане з мікроконтролером, а той, відповідно, з мікрокомп'ютером і сервером.

Абсолютно всі елементи, з яких складається даний стенд, дуже важливі і

виконують свої функції. Враховуючи, що інформації про існування подібних стендів знайти не вдалося, всі його складові, враховуючи характеристики та функції, були обрані самостійно.

3.2 Мікрокомп’ютер

Мікрокомп’ютер — це комп’ютер, який має маленькі розміри та неважку конструкцію. Так як розмір таких пристроїв в багато разів менший, ніж у звичайного ПК або ноутбуку, всі характеристики тут також слабкіші. Використовують мікрокомп’ютери тільки для виконання нескладних задач. Наприклад відображення на екрані найпростішої графіки.

Мікрокомп’ютери[4] зазвичай використовують мікропроцесори в ролі центрального, який і виконує як арифметичні, так і логічні операції. Їх відносять як до четвертого, так і до п’ятого покоління обчислювальних машин.

Одними з основних ознак мікрокомп’ютерів є:

1. Шинна організація системи;

2. Апаратні і програмні засоби повинні бути дотримані стандартам.

3. Розраховані на різні категорії користувачів.

Функціонал мікрокомп’ютерів, як і їх енергоспоживання, невисокий. Ціна цих пристроїв відповідна. А налаштувати їх можна тільки в сервісних центрах з допомогою ноутбуку або ПК.

Однією з переваг мікрокомп’ютерів є можливість нормальної роботи у складних умовах.

Для стенду було обрано мікрокомп’ютер (Рисунок 3.1) Orange Pi PC.

Цей пристрій має такі характеристики: 4-ядерний чіп Allwinner H3, 1 гігабайт оперативної пам’яті та графічний процесор GPU Mali-400MP2 600MHz. Також він має достатньо великий функціонал: HDMI, AV, Ethernet, 3 USB, microUSB-OTG та інфрачервоний порт для управління пультом. При потребі можна також підключити окремо модуль камери. З лівої сторони ще знаходиться отвір, до якого можна

підключити карту пам’яті MicroSD.

Після отримання на мікрокомп’ютер необхідно встановити операційну систему. Пристрій підтримує такі операційні системи як Android 4.4, Ubuntu, Debian Image, Armbian.

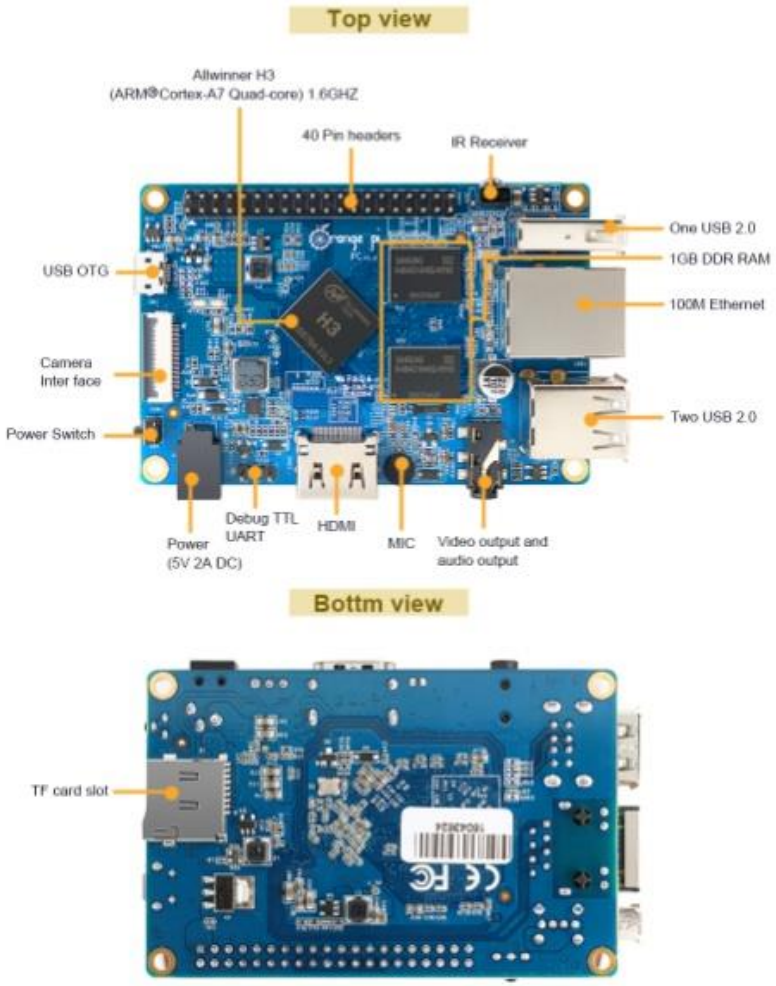

Рисунок 3.1 — мікрокомп’ютер Orange Pi PC

Зв’язок мікрокомп’ютера з мікроконтролером здійснюється з допомогою мікросхеми UART.

3.2.1 Налаштування мікрокомп’ютера

Перед початком роботи з мікрокомп’ютером потрібно підготувати програмне забезпечення для конкретної моделі.

Для роботи з Orange[12] Pi PC та його програмування необхідно завантажити та встановити операційну систему та додаткову бібліотеку. Для цього підходять операційна система Armbian та бібліотека WiringOP відповідно.

Armbian — примітивна операційна система, яка призначена для комп'ютерів, що складаються з однієї плати на базі ARM — процесорів. Вона створена на базі таких операційних систем як Ubuntu та Debian.

WiringOP — це бібліотека, яка необхідна для роботи з інтерфейсом GPIO(general-purpose input/output). Дана бібліотека є модифікацією WiringPi, яка спочатку була написана для Raspberry Pi. Компанія Orange Pi адаптували її для роботи зі своїми комп'ютерами та додали їй приставку OP, що означає Orange Pi.

3.3 Комп'ютерна шина

Комп'ютерною шиною називають багато сигнальних ліній, яким присутні деякі електричні характеристики. Використовуються для пересилання інформації від одного функціонального блока до іншого у комп'ютері. Є шини з послідовними та паралельними типами передачі сигналу або з асинхронними та синхронними. Також для внутрішнього призначення та інтерфейсні. Загалом в шин є різні розрядності, протоколи роботи, силою пропуску та пристроями, які вони підтримують.

Історично склалось, що шиною називали, зазвичай, інформаційні канали з приєднаними до них деякими пристроями, що забезпечували читання та відсилання інформації. Тому зараз так само говорять про з'єднання точка-точка.

Варто зазначити про спеціальні шини, наприклад шини у процесорах або які використовуються для зв'язку з відеоадаптерами.

Комп'ютерна шина застосовується також у материнських платах. А саме для підключення плат відео карти, модема та інших, а також для додаткових слотів.

Шини поділяються на 16-розрядні та 32-розрядні. Такої їх можна розділити на шини з високою продуктивністю та, відповідно, з низькою. Сучасні шини надають можливість приєднати до неї одночасно декількох пристроїв.

3.4 Мікросхема UART

Мікросхема UART — це такий тип асинхронного одночасно і приймача, і передавача. Він виконує передачу між формами послідовними і паралельними.

Майже завжди UART йде або у складі мікросхеми, або самостійною мікросхемою. Застосовується, щоб реалізувати підключення через периферійний або ж комп'ютерний порт. Сьогодні рахується, що UART входять до складу мікроконтролерів.

Також існує подвійний UART, його називають DUART. У ньому використовуються одночасно дві UART, поміщені на одну мікросхему

На теперішній час випускаються мікросхеми, які мають функцію комунікації саме у синхронному режимі. Їм дали назву USART.

UART включає такі складові:

1. Схема процесу прийому та передачі
2. Логіка для процесу запису і читання
3. Регістри виводу і регістри вводу
4. Тактовий генератор, в якого частота повинна бути кратна бітрейту.
5. Також можуть бути інші компоненти за потреби.

3.4.1 Послідовні прийом та передача

Завдяки такій системі, використовуючи, наприклад, дроти, біти даних можуть передаватися з місця на місце. Причому, якщо потрібно використовувати це на великих відстанях, тоді потрібно багато дротів і вартість збільшується. Тому, щоб уникнути цього, біти передаються по черзі. При цьому застосовують UART щоб на всіх кінцях лінії з'єднання перетворити паралельну у послідовну форму. Обов'язково повинні бути регістри, які й забезпечують можливість перетворення послідовних і паралельних форм.

UART не приймає, як правило, та не генерує сигнали, що пересуваються з

однієї частини обладнання, до іншої. Для того, щоб змінити логічний рівень сигналів та UART застосовується інтерфейсний блок.

Зовнішні сигнали здатні бути у різних формах. Відсутність або ж присутність струму застосовувалась колись у телеграфних схемах. Причому, частина з сигнальних схем працюють без дротів.

Існує 2 типи зв’язку: напівдуплексний та дуплексний. За першим типом, пристрої переходять від передачі до прийому та навпаки. Дуплексний зв'язок забезпечує функцію прийому і передачі разом.

Варто зазначити, що UART зазвичай застосовується в такому інтерфейсі, як RS-232, призначений для систем комунікацій. Для забезпечення зв'язку комп'ютера з мікроконтролером застосовується саме він. Існують також мікросхеми, які мають дешеву вартість, та які використовуються для перетворення в сигнал RS-232 логічного сигналу TTL.

3.4.2 Асинхронні прийом і передача

При передачі такого типу, UART спочатку відправляє початковий біт, після того 5-8 бітів з даними. Причому важливість першого найменша. Наступним йде біт парності, після якого 1, 1.5, або 2 так званих стопових біти. Полярність початкового біта повинна бути зворотною до стану ліній зв’язку. В свою чергу, стоповий біт ставить паузу перед тим, як підуть наступні дані. Біт парності визначає, скільки між непарними і парними та стоповими і стартовими одиниць. Також його може не бути. Якщо хоча б одиниця передалась, UART може пересинхронізуватись. В передачі синхронного типу біти стопу і старту не застосовуються. Це позитивно впливає на зв’язок ліній та передачі потрібних даних. У синхронному типі є необхідність підтримувати передачу даних постійно, щоб не пропав зв’язок. В асинхронної передачі такої проблеми немає.

3.5 Архітектура ARM

Архітектура ARM (Advanced RISC Machine) — це 32-бітна архітектура процесорів, що здебільшого використовується для створення портативних пристроїв. Технології, що допомагають заощаджувати енергію, це головна перевага та причина популярності цієї архітектури. Архітектура ARM застосована приблизно у 75% 32-бітних процесорів.

RISC — архітектура процесорів, у якій скорочений набір команд.

Архітектура ARM[3] має також певні властивості, які є і в RISC архітектурі:

1. Використовується завантажувально-зберігальна архітектура

2. Зміщений доступ до пам’яті також не підтримується

3. Для комфортного декодування та ціною значно меншої кількості коду, тут інструкція ширини 32 біти.

4. Виконання єдиним циклом

Щоб бути на рівні з іншими процесорами, у даній архітектурі були також додані нові можливості:

1. Менша кількість розходжень певних службових сигналів. Велика кількість інструкцій виконується умовно.

2. Умовні коди безпосередньо замінюють арифметичні інструкції лише у випадку, коли це необхідно.

3. Застосовується 32-бітна схема зсуву, яку можливо застосовувати в майже всіх арифметичних інструкціях

4. Значно вдосконалена адресна індексація

5. Щоб скоріше отримати доступ до списку функцій є регістр зв’язків.

Застосування усіх інструкцій тепер необов’язкове умовно. Адже в ARM з самого початку всіх інструкцій застосовується 4-бітний код обставини. Цікаво, що цей код в других архітектурах рахується як відгалуження. Завдяки цьому кодування біту, який повинен бути доступний для переміщення інструкції у вільну частину пам’яті, значною мірою скорочується.

Ще однією особливістю даних процесорів є служби на зразок як лічильник споріднених рішень, а також передприрістні методи адресації.

Кількість інструкцій в архітектурі ARM продовжує зростати. Це є доказом того, що вона з’явилась не так давно. Початкові процесори цієї фірми зовсім були позбавлені можливості генерувати такий код, котрий має бути введений методом очікування для С-об’єктів.

Цікаво також, що для процесорів ARM7 та старіших версій можна виділити 3 рівня конвеєру: загрузка, декодування і безпосереднє виконання. У новіших версіях пристрою, які мали більш кращі параметри, можна виділити 5 рівнів, а саме підсумовувач та вдосконалена передбачу вальна логіка. Це забезпечує кращу продуктивність.

У цій архітектурі існує 16 співпроцесорів. Кожен з них має номер відповідно від 0 до 15. Останній з них зберігається спеціально для певних контрольних функцій. Наприклад, управління кешом.

На пристроях з даною архітектурою периферія виконує з’єднання з процесором відповідно до карти регістрів в просторі співпроцесорів. Також можливе підключення до шини.

Для вдосконаленої компіляції кодів існує режим Thumb. Використовуючи його, процесор здатен здійснювати 16-розрядні інструкції. Майже всі з них змінюються на звичайні ARM.

В такому режимі чим менші коди, тим в них менше функціоналу: лише розгалуження мають функцію бути умовними.

3.6 Мікроконтролер

Мікроконтролер — це спеціальний, мікроелектронний пристрій, який можна запрограмувати. Він використовується здебільшого у вузлах управління деяких технічних виробів в різноманітних областях нашого життя, зокрема системах управління технологічними процесами.

Сфера використання мікроконтролерів дуже велика. Їх застосовують у всіх пристроях, починаючи з музичного пристрою і закінчуючи літаками та автомобілями.

Зазвичай такі пристрої мають маленьку розрядність(8 або 16 біт). Разом з цим в них є великий набір з команд, якими можна маніпулювати кожним бітом окремо. За допомогою цього можна управляти дискретним обладнанням. Наприклад увімкнути або вимкнути світло, зачинити або відчинити двері. Інструменти, за допомогою яких це стає можливо, називають бітовим процесором.

Однією з найважливіших особливостей мікроконтролера є наявність всіх, необхідних для створення системи управління, елементів у мікросхемі. Всередині мікроконтролера повинна бути як оперативна пам'ять, так і постійна. Також там мають бути присутні таймери та лічильники для командних циклів, генератор тактових імпульсів.

Найпростіша система, в основі якої лежить мікроконтролер, може складатися хоча б з самого мікроконтролера, зарядного блоку і декількох пасивних елементів.

При виробництві цих пристроїв, необхідно знати пропорції розміру та ціни, а також продуктивності разом з гнучкістю. Причиною цьому служить дуже велика кількість різних способів застосування, для яких важливі ці співвідношення. Велика різноманітність різних типів цих мікроконтролерів, які відрізняються між собою розміром, типом внутрішньої пам’яті, корпусом та архітектурою модуля.

Мікроконтролери, а саме 8-розрядного типу сьогодні активно застосовуються у багатьох галузях. Цікаво, що при цьому звичайні процесори такого ж розряду не мають місця бути. Їх замінили більш сильнішими моделями. Насправді це все має дуже просте пояснення. Є багато галузей, де просто не потрібно мати високу продуктивність. Тоді для чого купувати дорогі сильні процесори, якщо вся продуктивність його не буде використовуватись взагалі.

Компанії, що випускають мікроконтролери, звісно ж, хочуть щоб їх продукція працювала на високих частотах. Проте неможливість підвищити як ціну, так і енергоспоживання не дають змоги цього зробити. На практиці ж, ці компанії пропонують покупцям обирати різні версії, які відрізняються частотами і

використанням електроенергії. Існує велика кількість пристроїв, в яких для оперативної пам’яті і регістрів застосовується статична пам’ять. Через це з’являється змога мікроконтролера навіть при непрацюючому генераторі продовжувати працювати і навіть зберігати дані. Варто зазначити, що зазвичай є можливість переключати режими використання електроенергії контролером.

Багато типів контролерів не мають місця, куда можна підключити додаткову пам'ять. Проте існує так само багато різних мікроконтролерів, в яких присутні не тільки оперативна, а й незалежна пам’ять, яку можна використовувати для збереження різних даних. Виробництво таких контролерів має право існувати лише тоді, коли точно відомо, що програма для нього змінюватись більше не буде. Зручніші ж моделі, яким доступна функція перезапису даних в пам’ять.

Для реалізації апаратної частини стенду було обрано мікроконтролер(рисунок 3.2) Arduino[5] Mega 2560. Це пристрій, в основі якого використаний мікроконтролер ATmega2560.

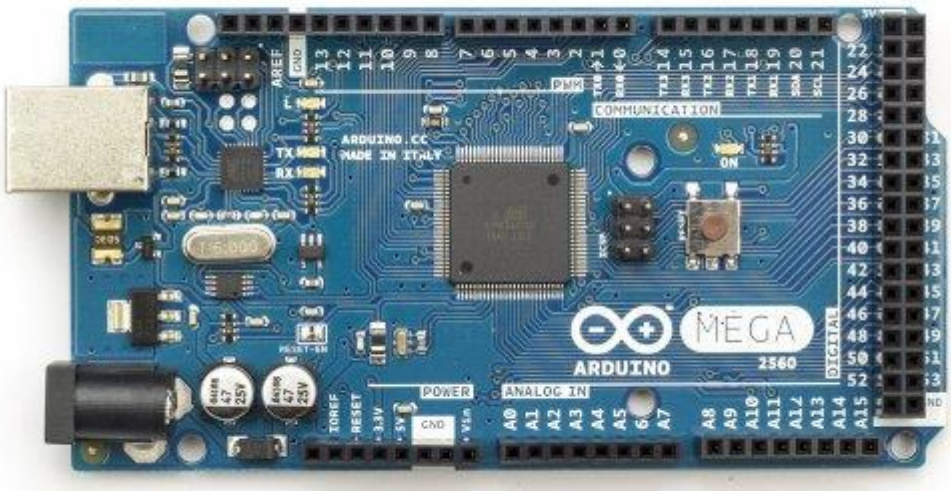

Рисунок 3.2 — мікроконтролер Arduino Mega 2560

Даний пристрій[13] складається з всього, що необхідне для комфортної роботи безпосередньо з самим мікроконтролером. А саме цілих 54 цифрових роз’єми, частина з яких має змогу використовуватися як виходи для ШІМ та 16 аналогових.

А також там є 4 UART та резонатор на 16 МГц, 3 входи для ICSP, живлення і, звісно ж, USB.

Всі технічні характеристики обраного мікроконтролера вказані у таблиці 3.1.

Робоча напруга	5 В
Flash-пам'ять	256 кб
Цифрові роз’єми	54
SRAM	8 кб
EEPROM	4 кб
Тактова частота	16 Мгц
Рекомендована напруга живлення	7-12 В
Максимальна напруга живлення	6-20 В
Максимальний ток одного виводу	40 мА
Максимальний вхідний ток виводу 3.3V	50мА

Таблиця 3.1 — технічні характеристики мікроконтролера

Почати роботу з цим мікроконтролером порівняно дуже легко. Варто лише підключити його до будь-якого джерела електроенергії. Адже для нього підходять більшість сучасних плат.

3.6.1 Програмування та захист

Мікроконтролер ATmega2560 продається відразу разом із вшитим загружчиком. Тому є можливість завантажувати будь-які програми без використання зовнішнього програматора відразу на пристрій.

Функція прошивання контролера використовуючи вхід для програмування ICSP(In Circuit Serial Programming)та без загружчика так само залишається актуальною.

Прошивка ATmega16U2/8U2 безпосередньо пов’язана з DFU-загружчиком(Device Firmware Update). Завдяки цьому, при оновленні прошивки пристрою не виникає жодних проблем.

Важливо зазначити також, що у даному мікроконтролері присутні запобіжники. Тому, захист USB-порту як від перегрузок, так і від коротких замикань забезпечений. Такі заходи обережності дійсно дають ще один рівень безпеки для порту. Адже комп'ютери в наш час здебільшого також мають захист від перегрузок. Тому, у випадку, коли на USB-порт буде йти ток 500 мА або більше, робота відразу ж припиниться. Відновити її можна буде лише після того, як усунеться причина проблеми.

3.7 Плата

Для даної роботи було обрано плату(Рисунок 3.3) Arduino Nano. Її особливістю є дуже маленький розмір. Це одна з найменших плат цього виробника. Саме її розмір є причиною тому, що вона використовується зазвичай у таких проектах, де компактність має велике значення.

Рисонок 3.3 — плата Arduino Nano

Дана плата[14] відрізняється від інших ще й тим, що тут відсутнє місце для підключення зовнішнього живлення. А працює вона використовуючи лише порт

miniUSB або microUSB.

Варто зазначити, що технічні характеристики(таблиця 3.2) цієї плати не гірші, ніж у конкурентів, розмір яких значно більший.

Найменування	Значення
Напруга живлення	5 В
Частота	16 МГц
Вага	7г
Кількість аналогових входів	8
Вхідне живлення	7-12 В
Максимальний ток цифрового входу	40 мА
Флеш-пам’ять	16 Кб або 32 Кб
Оперативна пам’ять	1 Кб або 2 Кб
Розміри	19х42 мм

Таблиця 3.2 — Технічні характеристики плати Arduino Nano

Плата[16] може заряджатись різними способами:

1. Використовуючи порти mini- та micro-USB при під'єднанні до джерела живлення.

2. За допомогою окремого джерела енергії, якщо його напруга в межах 6-20 В, а також якщо його пульсації відбуваються на низькому рівні

На цій платі також є певні межі, встановлені на її входи і виходи, призначені для напруги та току. Повністю всі, як аналогові, так і цифрові контакти мають такий робочий діапазон: 0-5 В. При порушенні цих меж, напруга повинна контролюватися спеціальними діодами, призначеними для захисту.

Дуже зручно організована система сигналів. На пристрої знаходяться 4 світодіоди. Вони призначені для індикації стану, в якому знаходиться в даний момент сигнал. Так, перший та другий світодіоди активні тоді, коли слабкий сигнал. Третій загорається лише у випадку, коли напруга 5 В. Він дає зрозуміти, що на

даний момент підключене живлення. І, відповідно, останній, четвертий світодіод зпрацьовує у випадку занадто високого сигналу.

3.8 Кінцеві модулі

Кінцеві модулі(Рисунок 3.4) — це модулі, які підключаються до мікроконтролера та реалізують увесь функціонал стенду. Стенд створений таким чином, що дана можливість підключити одночасно до 20 окремих модулів. Це рішення робить функціонал стенду дуже гнучким. Користувач може обрати той набір функцій, який йому потрібен і навпаки, відключити ті модулі, які його не цікавлять.

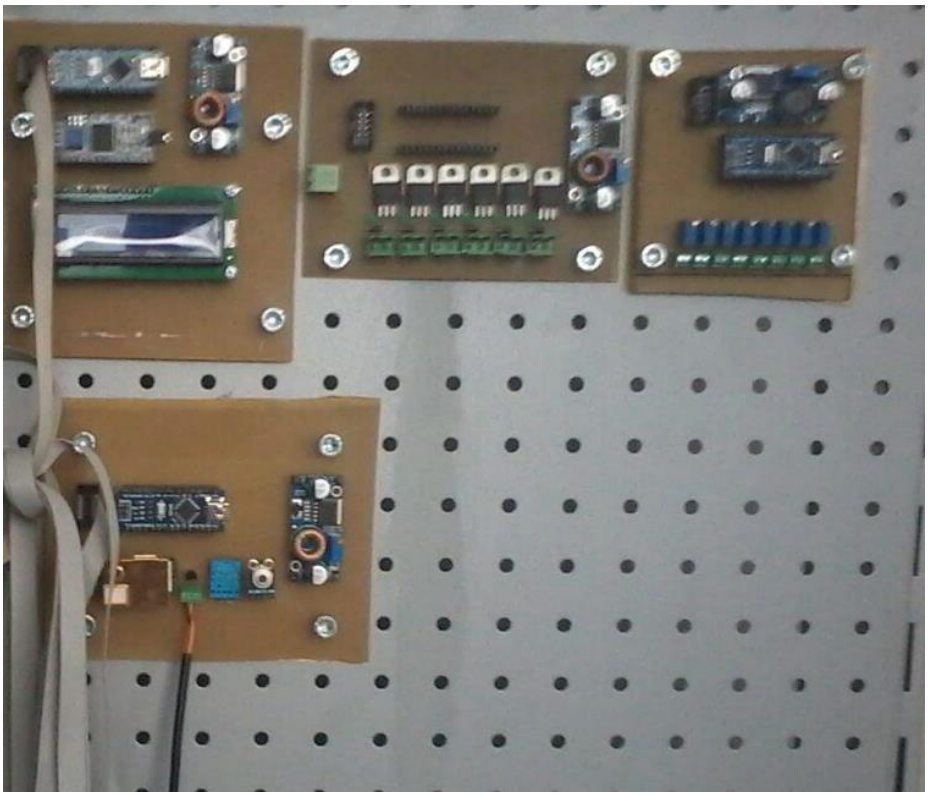

Рисунок 3.4 — кінцеві модулі

Кожен модуль відповідає за свої функції. Це може бути, наприклад, датчик вологості у кімнаті або температури повітря, освітленості або диму, руху або газу і багато інших. З допомогою додатку або web-сторінки завжди в реальному часі можна перевірити ті показники, що вас цікавлять. До того ж, реалізована функція

управління, тому також можна одним натиском на дисплей змінити, наприклад, рівень вологості або вимкнути світло у кімнаті.

Ще один елемент стенду — це маленький екран. Всі показники та команди також можуть бути відображені на ньому.

Мікроконтролер забезпечує комутацію усіх кінцевих модулів з мікрокомп’ютером, який, у свою чергу, ревізує web-інтерфейс та забезпечує зв'язок серверу зі стендом.

Висновки до розділу

У цьому розділі було проаналізовано весь стенд загалом та всі його елементи окремо. Розглянуті функції кожної з його складових. А також ознайомились з їх технічними характеристики і методами налаштування та програмування. Розглянуто схему роботи стенду.

4. ЗАСОБИ РОЗРОБКИ

Під час створення програмного продукту необхідно обрати технології, які будуть використовуватись в роботі. Було обрано середовище Microsoft Visual Studio 2017. Мовою програмування, в свою чергу, стала C++.

4.1 Опис середовища розробки Visual Studio 2017

Використовуючи Visual Studio, можна якісно і комфортно працювати. Однією з переваг цього середовища розробки є можливість рефакторингу та виправлення помилок безпосередньо в коді.

Зараз у Visual Studio[7] 2017 є IDE, редактор коду Visual Studio Code, який доступний для таких операційних систем як Mac, Windows і Linux. Також є сервіс Visual Studio Team Services для спільної роботи та Visual Studio Mobile Center для створення мобільних додатків.

Інструмент Microsoft вже встановили більше 21 000 000 разів, адже корпорація дійсно піклується про користувачів своєї продукції. Вони постійно намагаються вдосконалити свою продукцію та прислухаються до побажань розробників.

Сьогодні будь-хто, в кого є бажання, має змогу завантажити середовище розробки Visual Studio 2017. До того ж, він отримає 2-х місячну передплату до Xamarin University, де навчають розробці додатків для мобільних на мові C#.

Хочеться виділити вдосконалення та налагодження функції рефакторінгу для усіх мов, що підтримуються. З новою версією, з’явились і нові real-time методи перевірки залежностей і модульного тестування. Все це свідчить про те, що в Microsoft продовжують роботу та намагаються покращити умови для написання коду програмістами.

Зміни з’явилися і в процесі встановлення програми - тепер ви можете обрати лише ті компоненти, які вам потрібні. Процес установки став простішим та значно

швидшим загалом. Користувачам, також, тепер непотрібно створювати нові проекти для того, щоб налаштувати якусь необхідну частину коду.

Visual Studio розповіли на своїх презентаціях про інтеграцію платформи Azure. Microsoft цим надає змогу простіше налаштувати, опублікувати або зберегти ваші продукти у хмарі Azure разом із IDE. Вмонтовані інструменти, що допомагають працювати з додатками, .NET Core та іншими.

Необхідно також сказати, що сталися зміни у мобільній розробці. Користувачам надали вдосконалені інструменти генерації модульних тестів, а також для профілювання та налагодження. Створені дуже приємні умови для написання кроссплатформних продуктів, тому Visual Studio 2017 разом із Xamarin ідеально підійдуть для цього. Visual C++ для написання бібліотек тим же VS2017.

Visual Studio Mobile Center – це новий сервіс, який призначений здебільшого для розробників мобільних додатків. Він використовується для рішення тестування, моніторингу додатків та їх збірці. Зміни відбулися і тут. Корпорація розповіла про підтримку продуктів, які написані мовами програмування Java, Swift, ObjectiveC. Не забули й за Xamarin разом із React Native. Visual Studio Mobile Center Preview зараз можна спробувати абсолютно безкоштовно, до того ж там підтримується Espresso, є функція створення Distribution Groups і, як заявили, проведена робота над аналітикою.

Корпорація Microsoft, зокрема розробники VS також працюють над платформою для Mac – четверта версія продукту вже вийшла. Увага зосереджена на розробці мобільних додатків, хмарних рішень і, звісно, продуктів для операційної системи Mac. Підтримка відбувається також для NuGet та .NET Core. Оновлюються та вдосконалюються інструменти для створення мобільних додатків, виправляються помилки.

4.2 Особливості роботи у середовищі VS

Незалежно від того, якою мовою програмування ви користуєтесь, дане

середовище розробки значно допомагає у роботі.

IntelliSense тоді, коли відбувається ввід, описує API. Також тоді, коли відбувається автоматичне завершення, точність разом із швидкістю збільшуються. Присутня можливість перевірки API.

Поширеною проблемою при написанні достатньо великого коду є складність навігації. VS, за допомогою функції відображення визначення, значно спрощує це. Адже ви не губите початкову розмітку та контекст коду. Засіб GoTo також допомагає при роботі з великим кодом. Він фільтрує інформацію та вибирає необхідні елементи.

Ще однією можливістю є пошук всіх посилань. За допомогою цього можна зберігати будь-яку кількість результатів. До того ж, процес угрупування, пошуку результатів і їх фільтрації значно полегшується. Вдосконалена була також панель прокрутки. За допомогою цього можна шукати проблеми значно краще. Ви завжди будете розуміти, в якій саме частині коду ви знаходитесь завдяки функції візуалізації структури.

Знайти потрібні файли у вашому проекті можна доволі легко, використовуючи браузер рішень та структуру об’єктів.

Важливо добре орієнтуватися та розуміти код продукту. При роботі з CodeLens завжди є можливість дізнатися, чи змінювався метод та коли і ким були зроблені поправки в коді.

Помилки в коді допускають абсолютно усі. Але важливо знаходити їх та виправляти вчасно. Зазвичай це робиться на ходу, під час написання коду. Тому потрібно реагувати на них, виправити, замінити щось або виконати ре факторинг. Лампочки допоможуть з проблемами, які часто виникають у розробників при написанні продукту.

Неважливо, яка причина помилки. Для того, щоб здійснити перехід до них у рішенні та спробувати виправити їх, необхідний тільки їх список.

Деякі мови програмування використовують динамічні аналізатори, які ідентифікують проблеми домену у реальному часі.

Частіше всього, аби виправити помилку достатньо почитати рішення

проблеми в інтернеті. Для зручності, реалізована можливість пошуку інформації про проблему, виділивши помилку у коді та натиснувши F1.

Практика показує, що в залежності від того, як збільшується проект, виникає необхідність виконати рефакторинг коду, який писав хтось інший чи ви самі або його реструктуризацію. Такі мови програмування, як C++, VB, C# мають дуже корисні функції рефакторингу. Використовуючи меню Visual Studio Editor, можна швидко редагувати, виправляти, навіть витягувати метод та змінювати його назву.

Є ще одна особливість, якою не треба нехтувати, якщо вам це зручно. Макети для роботи з кількома моніторами можна зберігати та налаштовувати так, щоб вам було зручно та підходило для ваших екранів та виконання конкретних завдань.

4.3 Мова програмування C++

Мова програмування C++ є достатньо універсальною та входить до категорії високого рівня. Їй присутні об’єктно-орієнтоване, процедурне та інші парадигми програмування.

4.3.1 Технічний огляд мови

Дана мова[8] піддалась стандартизації ще у 1998 році. Виконала це Міжнародна організація стандартизації. Так, у тому році стандарт C++ можна було поділити на дві частини — стандартна бібліотека та ядро мови. До першої увійшла також бібліотека шаблонів STL, яка була розроблена в той же час, що й стандарт.

Сьогодні вже назва цієї бібліотеки не вживається в тому ж смислі, як тоді. Проте розробники і зараз використовують позначення STL. Вони так називають ту частину бібліотеки, в якій зберігається такі поняття як літератори, функтори, а також яка містить визначення шаблонів контейнерів.

Цікаво зазначити, що у стандарті мови C++ зазначене посилання, яке вказує на стандарт мови C від 1990 року. Функції, які були взяті із бібліотеки мови C, не визначені у стандарті мови C++.

Насправді, існує дуже багато інших бібліотек[9] мови С++, які не значаться у її стандарті. Так, працюючи з С++, є можливість застосовувати значну кількість бібліотек мови С.

Вважається, що стандартизація абсолютно чітко визначила мову С++. Насправді ж, там можуть бути елементи інших неповних мов програмування, які не піддались стандартизації, адже спочатку ця мова існувала окремо та розвивалась тоді, коли потрібно було виконати нове завдання. Одночасно розвивався компілятор під назвою Cfront.

Мова програмування С++, як і інші, має стандартну бібліотеку. Вона складається з стандартної бібліотеки мови С. Проте існують відмінності, необхідні для кращої роботи з іншою мовою. Але бібліотека мови С++ має також свою частину на базі STL. Там є такі необхідні інструменти, як ітератори та контейнери, куди входять такі поняття як списки, вектори та інші. З їхньою допомогою можна звертатися до контейнерів так само, як і до масивів. Також, використовуючи STL, з’являється можливість аналогічно працювати з контейнерами зовсім іншого типу: чергами, списками та іншими.

Без шаблонів неможливо було б використовувати узагальнені алгоритми, які здатні були б працювати з контейнерами чи навіть із послідовностями, в яких ітератори дають доступ безпосередньо до їх членів.

Використання можливостей бібліотек мови С++ здійснюється за допомогою директиви[11] #include. Ця дія активує стандартні файли, яких існує 50 штук.

STL спочатку була проектом фірми HP. Згодом нею займались SGI. І після цього вона була включена в стандарт мови С++, хоча і не використовуючи назву “STL”. Незважаючи на це, значна кількість програмістів продовжують її так називати для зручності, щоб виділяти її з стандартної бібліотеки. Оновленням STL займається спеціальний проект STLport. Проте він не єдиний. Є й інші проекти, які також розроблять інші застосування STL для різноманітних задач. Цікаво, що при виробництві компілятора для мови С++, неодмінно має бути дана бібліотека. Адже вона стала невід’ємною частиною стандарту і завжди активно використовується.

4.3.2 Відмінності від мови С

Одним із завдань при створенні мови С++ було залишити сумісність з такою мовою програмування, як С. Тому, майже всі програми, які були написані мовою С, можна спокійно запускати і вони будуть працювати з компілятором для С++. Варто зазначити, що синтаксис мови С також перейшов до мови С++.

Свої особливості мова С++ звісно ж має. Вона отримала чимало нових можливостей, недоступних у С.

1. С++, на відміну від С, підтримує ООП, використовуючи класи
2. Узагальнене програмування шаблонами
3. Розширена бібліотека
4. Нові типи даних
5. Можливість обробляти винятки
6. Додані простори імен
7. З’явились стандартні функції
8. Реалізовано перевантаження операторів та імен функцій

Мова С не має об’єктно-орієнтовних можливостей. А С++ надає їй такі функції. Так, були добавлені нові спеціальні класи, з допомогою яких можна реалізувати 3 основні властивості ООП. А саме поліморфізм, успадкування і, неодмінно, інкапсуляцію.

Варто зазначити, що деякі нові функції та можливості мови С++ були згодом добавлені і в мову С. Наприклад написання коментарів з допомогою символу “//” або оголошення циклу з допомогою “for”. І, навпаки, в новіших версіях мови С були добавлені свої можливості, недоступні для використання у мові С++. А саме вдосконалена робота із масивами-парамертами та впровадження макросів vararg.

У мові С++ з’явилось набагато більше різних функцій та можливостей, порівняно з С. Проте це не перешкоджає швидкості роботи програм, написаних на ній. Швидкість роботи майже не відрізняється.

4.3.3 Історія

Мова програмування C++ була створена ще у 1979 році данським програмістом Б’ярном Страуструпом. Свою теперішню назву мова отримала лише через 14 років після того. Одразу їй дали доволі логічну назву “C з класами”.

Мова C++ у 1990-х роках була надзвичайно популярною. Її використовували для різноманітних завдань. Наприклад розробка програмного забезпечення, відеоігор або серверних програм. Варто зазначити, що C++ також зробила свій внесок у формування таких популярних мов програмування як Java та C#.

Змінити назву на “C++” було ідеєю Ріка Масситті. Вона пішла від звичайного оператора мови C, що позначається як “++”. Також, популярним способом називати нову версію будь-якої програми було додавання до її назви приставки “+”.

Висновки до розділу

У цій частині були розглянуті мова програмування, середовище та засоби розробки. Також були проаналізовані відмінності мови та описані особливості роботи у Visual Studio 2017.

5. ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

В даному розділі описані способи подальшого використання стенду та реалізація його серверного застосунку. Описуються також інструменти роботи з ним та взаємодія між елементами стенду.

5.1 Архітектура системи

Програма для роботи з елементами стенду була написана на мові програмування C++ у середовищі розробки Visual Studio 2017. Ця система повинна виділяти та обробляти повідомлення, надіслані користувачем з додатку або спеціального web-сайту. Зв’язок користувача зі стендом забезпечує мікрокомп’ютер. Після обробки повідомлення система визначає тип повідомлення, в разі знаходження там запиту, здійснюється зв’язок мікрокомп’ютера з кінцевими модулями за допомогою мікроконтролера. До них надсилається відповідний запит та очікується відповідь. Мікроконтролер зчитує дані з кінцевих модулів та передає їх назад на мікрокомп’ютер і той, в свою чергу, формує результат та виводить повідомлення на екран користувача. Наприклад, користувач надіслав запит про поточну температуру повітря.

У разі ж, якщо у повідомленні користувача був не запит, а команда, мікроконтролер також забезпечує комутацію мікрокомп’ютера з кінцевими модулями. Але тепер система розуміє, що це команда та відбувається відповідна дія.

Після закінчення, на мікрокомп’ютер передається повідомлення про закінчення, формується результат і виводиться на екран користувача. Наприклад, користувач надіслав команду вимкнути світло.

Для більшої зрозумілості принципу роботи програми, враховуючи її архітектуру було створено діаграму(рисунок 5.1 та 5.2) класів. На ній показані необхідні класи та їх відношення.

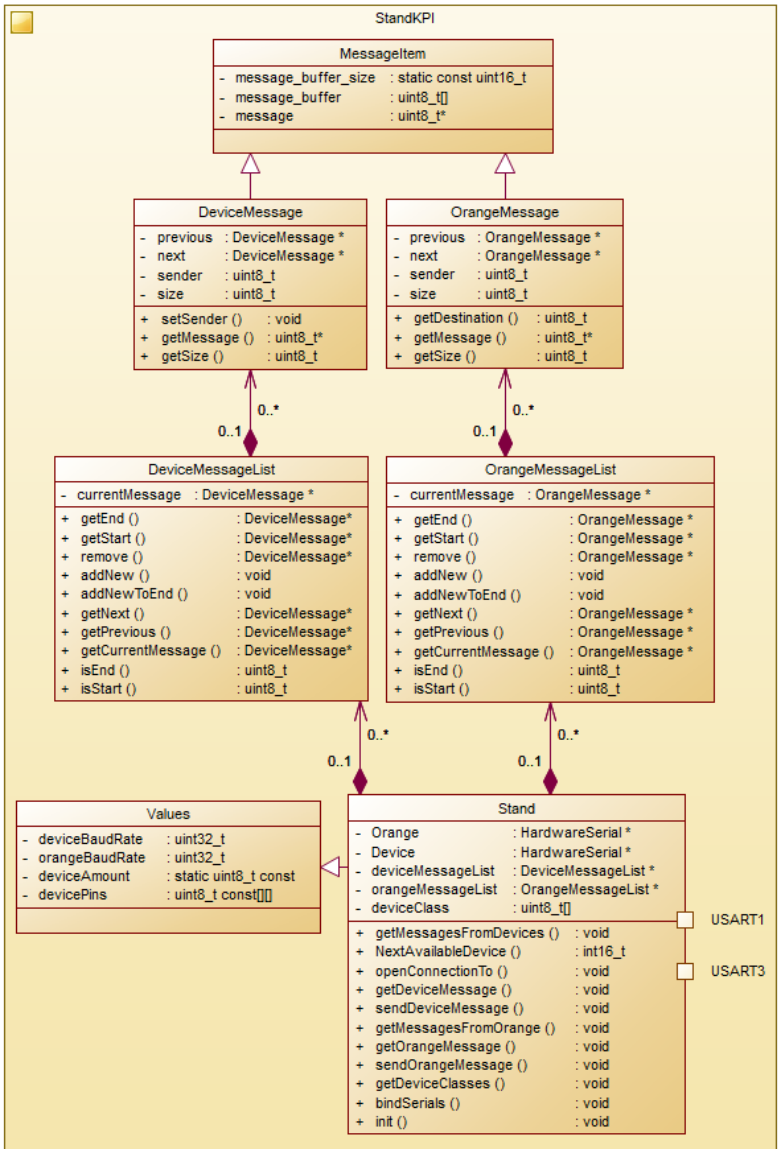

Рисунок 5. 1 — діаграма класів

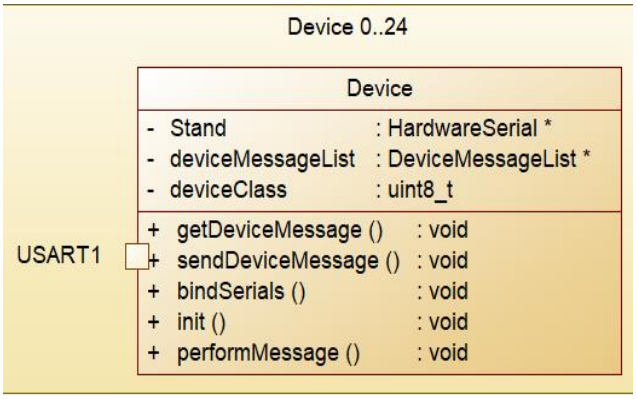

Рисунок 5. 2 — діаграма класів

5.2 Опис класів

Клас Stand можна визначити як об’єкт-обгортка взаємодії кінцевих модулів та web-інтерфейсу. Він включає в себе такі класи як Values, DeviceMessageList та OrangeMessageList. Тобто цей клас реалізує зв’язок та передачу даних кінцевих модулів з сервером.

Ще одним класом є MessageItem. Він є абстрактним класом та представляє архітектуру повідомлення.

Кінцеві модулі, що забезпечують весь функціонал системи, надсилають дані до користувача. Клас DeviceMessage є об’єктом, який це реалізовує. Він приймає повідомлення з них та утримує їх.

В свою чергу є клас DeviceMessageList, який включає попередній клас DeviceMessage, обробляє повідомлення та формує їх список.

Щоб користувачу отримати результат, потрібно надіслати запит на стенд. Ці запити отримує клас OrangeMessage та зберігає їх.

Аналогічно, є клас OrangeMessageList, який включає OrangeMessage і також обробляє запити та формує список.

5.3 Вивід даних

Щоб користувач міг користуватися даною системою, вихідні дані стенду потрібно вивести. У цій роботі передбачено 2 способи реалізації цього:

1. вивід на вмонтований екран
2. вивід на екран будь-якого пристрою за допомогою web-сайту

Програма виводу даних на екран(рисунок 5.2) написана також мовою C++ і може виводити на екран стенду результати запитів або виконання команд. А також, є можливість вивести на екран будь-яке повідомлення, введене користувачем за допомогою додатку.

Вивід результату за допомогою додатку або інтерфейсу(рисунок 5.3) web-сторінки була реалізована у другій частині цієї дипломної роботи.

Рисунок 5.2 — екран стенду

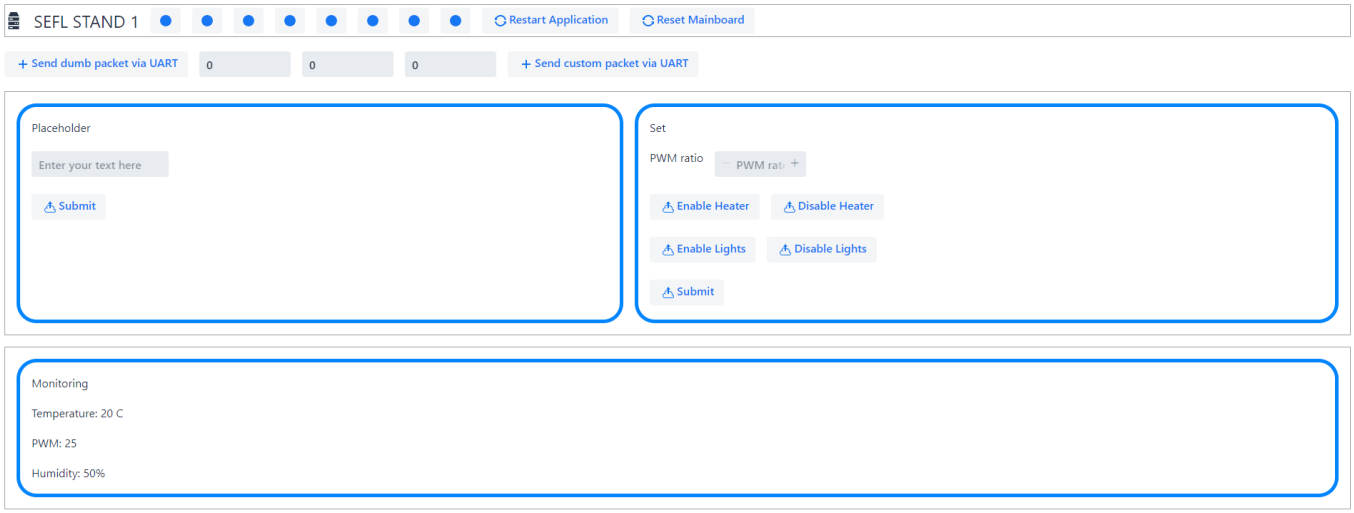

Рисунок 5.3 — інтерфейс web-сторінки

На інтерфейсі реалізовані поля вводу для команд або виводу тексу на екран. Також тут є функції переключати режими певних показників. Знизу також виводяться показники, взяті з кінцевих модулів.

Висновки до розділу

В цьому розділі було розглянуті архітектура програми, програмна реалізація

роботи. Також описані класи програми. Показані результати виводу результатів.

ВИСНОВКИ

Під час виконання даної роботи було розроблено проект стенду для моделювання віддаленого моніторингу та управління технологічними процесами.

Куплено усі необхідні складові: мікрокомп’ютер Orange Pi PC, мікроконтролер Arduino Mega 2560, плата Arduino Nano, екран для виводу повідомлень і результатів та кінцеві модулі.

Стенд був зібраний та підключений за певним принципом. Використовувалась модульна архітектура для більшої універсальності. Передбачена можливість підключення нових кінцевих модулів або відключення непотрібних. Всього можливо підключити 20 таких модулів. Причому всі можуть працювати одночасно.

Мікрокомпютер ревізує web-інтерфейс та забезпечує зв’язок користувача зі стендом.

Мікроконтролер забезпечує комутацію мікрокомп’ютера з кінцевими модулями

Кінцеві модулі забезпечують весь функціонал стенду.

Серверна частина була реалізована за допомогою мови програмування C++.

Реалізована можливість моніторингу та управління.

Користувач має доступ до всього функціоналу стенду за допомогою будь-якого пристрою, що має доступ до інтернету. Він може подивитися показники з усіх підключених кінцевих модулів, зайшовши на сайт. Також є можливість змінювати параметри у режимі реального часу.

Отриманий стенд максимально наглядно демонструє принцип роботи систем віддаленого моніторингу та управління. Зокрема розумного будинку.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Home Automation and Wiring//McGraw-Hill Professional / James Gerhart., 31. – 322 с.

2. Mann William C. The state of the science//Smart technology for aging, disability and independence. / Mann William C., 24. – 379 с.

3. Микроконтроллеры AVR. Практикум для начинающих / Вячеслав Хартов., 2012. – 278 с.

4. Мікрокомп'ютер Raspberry Pi - інструмент дослідника / С. Могильний., 2014. – 340 с.

5. Arduino и Raspberry Pi в проектах Internet of Things / Виктор Петин., 2016. – 320 с. – (БХВ-Петербург).

6. Arduino, датчики и сети для связи устройств / Том Иго., 211. – 544 с.

7. Microsoft Visual Studio [Електронний ресурс] – Режим доступу до ресурсу: https://www.visualstudio.com.

8. Visual Studio C++ [Електронний ресурс] – Режим доступу до ресурсу: https://docs.microsoft.com/en-us/cpp/get-started/tutorial-console-cpp?view=vs-2019.

9. Лафоре Р. Объектно-ориентированное программирование в C++. Классика Computer Science / Роберт Лафоре., 2018. – 928 с.

10. Прата С. Язык программирования C++. Лекции и упражнения / Стивен Прата., 2016. – 1244 с.

11. Вандевурд Д. Шаблоны C++. Справочник разработчика / Д. Вандевурд, Н. Джосаттис, Г. Дуглас., 2016. – 848 с. – (2-е).

12. Orange Pi [Електронний ресурс] – Режим доступу до ресурсу: http://www.orangepi.org/orangepipc/.

13. Arduino [Електронний ресурс] – Режим доступу до ресурсу: https://www.arduino.cc/en/Guide/ArduinoMega2560.

14. Белоус А. Основы технологии микромонтажа интегральных схем / А. Белоус, В. Емельянов., 2013. – 318 с.

15. Бобух А. Автоматизовані системи керування технологічними процесами: Навч. посібник. / А. Бобух. – Харків: ХНАМГ, 2006. – 185 с.

16. Гусев В. Электроника и микропроцессорная техника / В. Гусев. – Москва, 2005. – 790 с.

ДОДАТОК 1

Серверний застосунок стенду для моделювання віддаленого моніторингу та управління технологічними процесоми

Специфікація

КР.НТУУ”КПІ ім. Ігоря Сікорського”_ТЕФ_АПЕПС_ТР5152_19Б

Аркушів 1

Київ 2019

Позначення	Найменування	Примітки
Документація		
УКР.НТУУ” КПІ ім. Ігоря Сікорського”_ТЕФ_АПЕПС_ТР5152_19Б	Записка.docx	Пояснювальна записка
Компоненти		
УКР.НТУУ” КПІ ім. Ігоря Сікорського”_ТЕФ_АПЕПС_ТР5152_19Б 12-1	DeviceMessage.cpp DeviceMessageList.cpp MessageItem.cpp OrangeMessage.cpp OrangeMessageList.cpp Stand.cpp Values.cpp	Основні компоненти
УКР.НТУУ” КПІ ім. Ігоря Сікорського”_ТЕФ_АПЕПС_ТР5152_19Б 12-2	Додаток 2.doc	Опис програмного модуля

ДОДАТОК 2

Серверний застосунок стенду для моделювання віддаленого моніторингу та управління технологічними процесами

Лістинг програмного модулю

КР.НТУУ”КПІ ім. Ігоря Сікорського”_ТЕФ_АПЕПС_ТР5152_19Б

Аркушів 7

Київ 2019

```
/*
 * DeviceMessage.cpp
 *
 *  Created on: May 16, 2019
 *      Author: curab
 */
#include "DeviceMessage.h"
DeviceMessage::DeviceMessage() {
	// TODO Auto-generated constructor stub
	MessageItem();
	this->sender = -1;
	this->next = NULL;
	this->previous = NULL;
	this->size = -1;
}
void DeviceMessage::setSender(uint8_t sender) {
	this->sender = sender;
}
uint8_t* DeviceMessage::getMessage() {
	uint16_t size = 2;
	while (size < this->message_buffer_size && this->message_buffer[size - 1]
!= 0) {
		size = this->message_buffer[size - 1] + size + 1;
	}
	delete[] this->message;
	this->message = new uint8_t[size + 1];
	this->size = size + 1;
	this->message[0] = this->sender;
	for (uint16_t i = 1; i <= size; ++i) {
		this->message[i] = this->message_buffer[i - 1];
```

```
	}
	return this->message;
}
uint8_t DeviceMessage::getSize() {
	return this->size;
}
DeviceMessage::~DeviceMessage() {
	// TODO Auto-generated destructor stub
	delete[] this->message;
}

/*
 * DeviceMessageList.cpp
 *
 *  Created on: May 21, 2019
 *      Author: curab
 */
#include "DeviceMessageList.h"
#include <stdint.h>
#include "DeviceMessage.h"
DeviceMessageList::DeviceMessageList() {
	// TODO Auto-generated constructor stub
this->currentMessage=NULL;
}
DeviceMessage* DeviceMessageList::remove() {
	if (this->currentMessage == NULL) {
		return this->currentMessage;
	}
	if (this->isEnd()) {
		delete this->currentMessage;
```

```
            return NULL;
        }
        DeviceMessage* temp;
        temp = this->currentMessage->next;
        if (this->isStart() == false) {
            this->currentMessage->previous->next = temp;
        } else {
            temp->previous = NULL;
        }
        delete this->currentMessage;
        this->currentMessage = temp;
        return temp;
}
void DeviceMessageList::addNew(DeviceMessage *temp) {
        if (this->currentMessage != NULL) {
            temp->previous = this->currentMessage;
            temp->next = this->currentMessage->next;
            if (this->isEnd() == false) {
                this->currentMessage->next->previous = temp;
            }
            this->currentMessage->next = temp;
        } else {
        }
        this->currentMessage = temp;
        //this=temp;
        //return temp;
}
DeviceMessage* DeviceMessageList::getEnd() {
        //DeviceMessage* temp = this->currentMessage;
        if (this->currentMessage == NULL) {
```

```
                return NULL;
        }
        while (this->isEnd() == false) {
                this->currentMessage = this->currentMessage->next;
        }
        return this->currentMessage;
}
DeviceMessage* DeviceMessageList::getStart() {
        //DeviceMessage* temp = this;
        if (this->currentMessage == NULL) {
                return NULL;
        }
        while (this->isStart() == false) {
                this->currentMessage = this->currentMessage->previous;
        }
        return this->currentMessage;
}
void DeviceMessageList::addNewToEnd(DeviceMessage* temp) {
                this->getEnd();
                this->addNew(temp);
}
uint8_t DeviceMessageList::isEnd() {
        if (this->currentMessage->next == NULL) {
                return true;
        }
        return false;
}
uint8_t DeviceMessageList::isStart() {
        if (this->currentMessage->previous == NULL) {
                return true;
```

```
        }
        return false;
}
DeviceMessage* DeviceMessageList::getCurrentMessage() {
        return currentMessage;
}
DeviceMessage* DeviceMessageList::getNext() {
        return this->currentMessage=this->currentMessage->next;
}
DeviceMessage* DeviceMessageList::getPrevious() {
        return this->currentMessage=this->currentMessage->previous;
}
DeviceMessageList::~DeviceMessageList() {
        // TODO Auto-generated destructor stub
}

/*
 * MessageItem.cpp
 *
 *  Created on: May 16, 2019
 *      Author: curab
 */
#include "MessageItem.h"
MessageItem::MessageItem() {
this->message=NULL;
}
MessageItem::~MessageItem() {
        // TODO Auto-generated destructor stub
}
```

```
/*
 * OrangeMessage.cpp
 *
 *  Created on: May 16, 2019
 *      Author: curab
 */
#include "OrangeMessage.h"
OrangeMessage::OrangeMessage() {
	// TODO Auto-generated constructor stub
	MessageItem();
	this->next=NULL;
	this->previous=NULL;
	this->size=-1;
}
uint8_t OrangeMessage::getDestination() {
return this->message[0];
}
uint8_t* OrangeMessage::getMessage() {
	uint16_t size=3;
	while(size < this->message_buffer_size && this->message_buffer[size-1]!=0){
		size=this->message_buffer[size-1]+size+1;
	}
	delete[] this->message;
	this->message = new uint8_t[size-1];
	this->size=size-1;
	for(uint16_t i=0;i<size-1;++i){
		this->message[i]=this->message_buffer[i+1];
	}
	return this->message;
```

```
}
uint8_t OrangeMessage::getSize() {
	return this->size;
}
OrangeMessage::~OrangeMessage() {
	// TODO Auto-generated destructor stub
}
```

ДОДАТОК З

Серверний застосунок стенду для моделювання віддаленого моніторингу та управління технологічними процесами.

Опис програмного модулю

КР.НТУУ”КПІ ім. Ігоря Сікорського”_ТЕФ_АПЕПС_ТР5152_19Б

Аркушів 8

Київ 2019

Анотація

Розділ містить опис частини, яка слугує для забезпечення прийому повідомлень, команд або запитів від користувача та їх обробка. Та навпаки, система зчитує дані з кінцевих модулів та відправляє їх користувачу. Так здійснюється моніторинг показників та управління функціоналом стенду.

ЗМІСТ

Загальні відомості

У додатку міститься частина коду, яка забезпечує виконання функцій прийому та обробки повідомлень.

Програма була написана мовою C++ у середовищі розробки Visual Studio 2017.

ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ

У даній програмі реалізуються такі функції:

1. Прийом повідомлення
2. Утримання цього повідомлення
3. Класифікація повідомлення
4. Передача у відповідний модуль.

ОПИС ЛОГІЧНОЇ СТРУКТУРИ

Модуль реалізований у формі класу. Він забезпечує зв’язок між користувачем та кінцевими модулями, отримуючи дані та відправляючи їх на відповідні модулі у реальному часі. Тобто реалізує віддалений моніторинг.

ТЕХНІЧНІ ЗАСОБИ ЩО ВИКОРИСТОВУЮТЬСЯ

Модуль розроблено у середовищі розробки Microsoft Visual Studio 2017. При роботі програми використовуються такі пристрої як мікрокомп’ютер Orange Pi PC, мікроконтролер Arduino Mega 2560, плата Arduino Nano, та кінцеві модулі.

ВИКЛИК І ЗАВАНТАЖЕННЯ

Система забезпечує віддалений моніторинг в реальному часі, тому цей модуль запускається автоматично після запуску додатку і повертає користувачу оброблені дані з кінцевих модулів.

ВХІДНІ І ВИХІДНІ ДАНІ

Вхідними даними є інформація, яку надсилає користувач через додаток. Це може бути запит, команда або звичайний текст.

Вихідними ж є оброблені кінцевими модулями результати показників датчиків